**ABSTRACT:**

Lung cancer is a pervasive global health challenge, claiming numerous lives annually and ranking among the leading causes of cancer-related deaths worldwide. Early detection of lung cancer is paramount, as it significantly improves patient outcomes and survival rates. Medical imaging, including X-rays and CT scans, plays a pivotal role in early diagnosis. However, manual interpretation of these images is time-consuming and prone to human error. To address this issue, there is an urgent need for a robust and efficient image processing system capable of automating lung cancer detection and classification.

This project aims to create an advanced image processing system for the autonomous detection and classification of lung cancer in medical images. The primary objectives include image preprocessing, region localization, feature extraction, and classification into benign or malignant cases. Challenges encompass data variability, minimizing false results, and real-time processing.

The proposed solution, lung analysis, is an image processing script that employs a multifaceted approach. It begins with preprocessing to enhance image quality and mitigate noise. Morphological operations and active contour segmentation are applied to identify regions of interest. Circular feature detection is utilized to pinpoint potential nodules or masses. The system's outcomes promise to revolutionize the speed and precision of lung cancer diagnoses, reducing the burden on radiologists and contributing to global healthcare improvements.

Lung analysis offers a promising step towards the early detection of lung cancer, with the potential to save lives and improve healthcare outcomes. However, its efficacy depends on the quality of input images and may benefit from further validation and optimization for specific clinical applications.

**BACKGROUND:**

Lung cancer stands as a formidable global health challenge, claiming numerous lives every year and ranking among the leading causes of cancer-related deaths worldwide. Timely detection and intervention are pivotal in improving patient outcomes, as they enable early treatment and an increased chance of survival. Among the various methods used for early detection, the analysis of medical images, such as X-rays and CT scans, plays a critical role. However, manual interpretation of these images is time-consuming, and it is susceptible to human error. To address these issues, there is a pressing need for a robust and efficient image processing system capable of automating the detection and classification of lung cancer from medical images.

The Significance of Early Detection:

Early detection of lung cancer is of paramount importance. At later stages, the disease becomes more aggressive and challenging to treat. Early intervention allows for the initiation of suitable treatment plans, potentially increasing the chances of survival and reducing the severity of treatments required. Automating the process of lung cancer detection in medical images is a promising approach to achieve these goals.

**PROBLEM DEFINITION:**

In response to the urgent imperative for early lung cancer detection, this project is dedicated to the creation of an advanced image processing system designed to autonomously detect and classify lung cancer from a multitude of medical images, including X-rays and CT scans. The central objectives of this initiative encompass a multifaceted approach, commencing with image preprocessing to standardize formats, enhance image quality, and mitigate noise, followed by the precise localization of pertinent regions within the images. Once these regions are accurately identified, the system will employ sophisticated feature extraction techniques to dissect size, shape, texture, and intensity, subsequently enabling the critical classification of these regions as either benign or malignant. While navigating this journey towards improved healthcare outcomes, the project faces significant challenges, including the management of data variability across different sources, the imperative to minimize the occurrence of false positives and false negatives, and the crucial need for real-time processing. Successful realization of this endeavor promises to revolutionize the speed and precision of lung cancer diagnoses, alleviate the burdens placed on radiologists, and make a profound contribution to the worldwide initiative of harnessing the power of artificial intelligence for the betterment of healthcare applications.

**PROPOSED SOLUTION:**

Lunganlaysis is an image processing script aimed at analyzing medical images of the lungs to identify potential abnormalities, such as nodules or masses.

1. Reading and Displaying the Image:

- The code reads an input image named 'Lung1.jpg' and displays it in the first subplot. This subplot is labeled "Selected Image."

2. Noise Removal:

- The input image is converted to grayscale using `rgb2gray`.

- Noise is reduced using a 3x3 average filter (`fspecial('average', 3)`), and the result is divided by 255 to normalize pixel values.

- The denoised image is displayed in the second subplot labeled "Noise Removed."

3. Binary Image Conversion:

- The denoised grayscale image is converted into a binary image using a threshold of 0.2.

- The binary image, which simplifies the image into black and white regions, is displayed in the third subplot labeled "Binary Image."

4. Morphological Opening:

- Two different flat, disk-shaped structuring elements (`se1` and `se2`) are created with radii of 2 and 8, respectively.

- Morphological opening is applied to the binary image using these structuring elements. This operation can help separate connected regions in the image.

- The results of morphological opening for both structuring elements are displayed in separate subplots with the title "Opening Image."

5. Inverted Images:

- Two inverted binary images are created by subtracting the morphological opening results from an image filled with ones. These inverted images can highlight potential regions of interest.

- The two inverted images are displayed in separate subplots with the title "Inverted Picture."

6.Image Segmentation:

- A binary mask (`mask`) is specified to define the initial contour for segmentation.

- The `active contour` function is used to perform image segmentation with 800 iterations for the first inverted image and 400 iterations for the second inverted image.

- The segmented images are combined with their respective inverted images to create new images (`mix\_Image\_1` and `mix\_Image\_2`).

- Median filtering is applied to these combined images to reduce noise and improve image quality.

7.Final Binary Images

- The filtered combination images are thresholded with a value of 0.6 to create final binary images.

- The final binary images are displayed in separate subplots with the title "Segmented" and "Final Picture."

8. Circle Detection:

- The code detects circles within one of the final binary images using the `imfindcircles` function.

- The detected circles are displayed in green using the `viscircles` function, and the number of detected circles is shown.

- The image with detected circles is displayed in a subplot titled "Circle Found Picture."

9. Segmented Circles:

- A segmented image is created by subtracting one of the final binary images from the other. This operation helps identify specific circular regions of interest.

- The code further processes this segmented image to create a binary image.

- Boundaries of segmented circular regions are detected, and they are plotted in green in a subplot labeled "Circles Segmented."

The code is designed to preprocess and analyze lung images, with a focus on identifying potential nodules or circular abnormalities. The script combines various image processing techniques, thresholding, and contour-based segmentation to isolate regions of interest within the lung images. The detection of circular features may be useful in identifying potential nodules, which is important for early lung cancer detection.

**CODE:**

function lunganlaysis(Lung1\_jpg)

%Reads Image Files

Selected\_Image = imread('Lung1.jpg'); %read picture

subplot(3,3,1); %divides figure into rectangular panes

imshow(Selected\_Image); %show image

title('Selected Image'); %image name

%Noise Removal Section

greyscale\_Method = rgb2gray(Selected\_Image); %convert into grayscale

median\_filtering\_Image = filter2(fspecial('average',3),greyscale\_Method)/255; %Uses Noise Removal

subplot(3,3,2); %divides figure into rectangular panes

imshow(median\_filtering\_Image); %show image

title('Noise Removed'); %image name

%Binary Image Conversion

binary\_picture = im2bw(median\_filtering\_Image, 0.2); %Convert into Black and White

subplot(3,3,3); %divides figure into rectangular panes

imshow(binary\_picture); %show image

title('Binary Image'); %image name

%Create morphological structuring element

se1 = strel('disk', 2); %creates a flat, disk-shaped structure,

postOpenImage\_1 = imopen(binary\_picture, se1);

subplot(3, 3, 4); %divides figure into rectangular panes

imshow(postOpenImage\_1); %show image

title('Opening Image'); %image name

%Create morphological structuring element

se2 = strel('disk', 8); %creates a flat, disk-shaped structure,

postOpenImage\_2 = imopen(binary\_picture, se2);

inverted = ones(size(binary\_picture));

%Creates Inverted Picture

invertedImage\_1 = inverted - postOpenImage\_1;

invertedImage\_2 = inverted - postOpenImage\_2;

subplot(3,3,5); %divides figure into rectangular panes

imshow(invertedImage\_1); %show image

title('Inverted Picture'); %image name

%Specify initial contour

mask = zeros(size(invertedImage\_1));

mask(50:end-50,50:end-50) = 1;

%Segments image into foreground and background

bw\_1 = activecontour(invertedImage\_1, mask, 800); %800 iterations

bw\_2 = activecontour(invertedImage\_2, mask, 400); %400 iterations

%Create Combination Pictures with Inverted image and Contour

mix\_Image\_1 = invertedImage\_1 + bw\_1;

filter\_mix\_Image\_1 = medfilt2(mat2gray(mix\_Image\_1),[5 5]); %Filtering

mix\_Image\_2 = invertedImage\_2 + bw\_2;

filter\_mix\_Image\_2 = medfilt2(mat2gray(mix\_Image\_2),[7 7]); %Filtering

%Black White Combination to Create Final Images

final\_2 = im2bw(filter\_mix\_Image\_2, 0.6);

final\_1 = im2bw(filter\_mix\_Image\_1, 0.6);

pre\_final = final\_1; %transfer

subplot(3,3,6); %divides figure into rectangular panes

BW5 = imfill(pre\_final,'holes');

imshow(BW5); %show image

title('Segmented'); %image name

%Dispaly Final Image

subplot(3,3,7); %divides figure into rectangular panes

imshow(final\_1); %show image

title('Final Picture'); %image name

%Find Circles within Image using Polarity and Sensitivity

warning('off', 'all')

circle\_image = final\_1;

[centers,radii] = imfindcircles(circle\_image,[1 9],'ObjectPolarity','dark','Sensitivity',0.88);

viscircles(centers,radii,'EdgeColor','g'); % Circles Display Green

display(size(centers, 1), ' Numbers of Circles');

subplot(3,3,8); %divides figure into rectangular panes

imshow(circle\_image); %show image

title('Circle Found Picture'); %image name

segment\_pic = final\_2 - final\_1; % Creates Segmented Image

%Creates before Colour Image

pre\_colour\_pic = im2bw(medfilt2(mat2gray(segment\_pic),[3 3]), 0.6);

subplot(3,3,9); %divides figure into rectangular panes

imshow(pre\_colour\_pic); %show image

title('Circles Segmented'); %image name

%Circles Segmented Circles

[B] = bwboundaries(pre\_colour\_pic,'holes');

hold on

for k = 1:length(B)

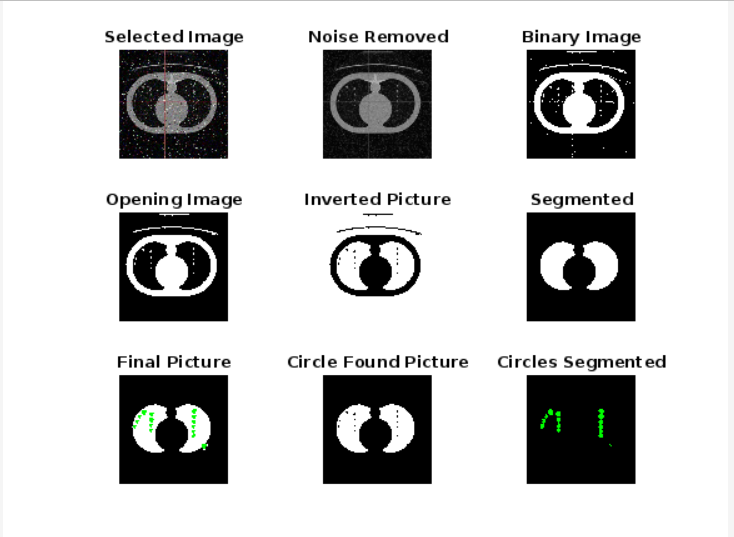
boundary = B{k};

plot(boundary(:,2), boundary(:,1), 'g', 'LineWidth', 2)

end

end

**SNAPSHOTS:**

****

**CONCLUSION:**

In conclusion, the code provides a comprehensive image processing workflow for the analysis of lung images. It combines various techniques to preprocess, segment, and detect potential abnormalities, with a particular focus on circular features that might be indicative of lung nodules or masses. The results can assist in the early detection of lung-related abnormalities, contributing to medical diagnostics and patient care. However, the effectiveness of the code may depend on the quality and characteristics of the input lung images and may require further validation and optimization for specific applications.